UNITED STATES PATENT APPLICATION

OF

DOUGLAS BOGIA et al.

FOR

SELECTING A DEVICE DRIVER FOR A PERIPHERAL DEVICE ADAPTED TO OPERATE ON A NETWORK AND SIMPLIFYING SECONDARY PRINTER INSTALLATION

Prepared by:

Pillsbury Madison & Sutro LLP

INTELLECTUAL PROPERTY GROUP

SELECTING A DEVICE DRIVER FOR A PERIPHERAL DEVICE ADAPTED TO OPERATE ON A NETWORK AND SIMPLIFYING SECONDARY PRINTER INSTALLATION

BACKGROUND OF THE INVENTION

Field of the Invention

This invention relates to configuring a computer for installation of a peripheral device. In particular, device drivers are selected for computer peripherals, based upon an identification code associated with the peripheral.

Description of the Related Art

Most personal computer (PC) systems facilitate the installation of newly connected hardware devices by first detecting that the new device has been connected to a bus of the computer. For example, the Microsoft® Windows® "Install New Hardware Wizard," may be activated by a user to assist in detecting and identifying the newly connected hardware. The Hardware Wizard recognizes newly connected hardware, and guides the user through the installation of the correct device drivers. A device driver is a software program that matches standardized commands from the computer's operating system to specific capabilities of the newly connected device. To aid in this process, Microsoft® Windows®, for example, uses a popular software routine called Plug and Play.

Plug and Play is a computer implemented routine that provides a host PC with an ability to detect the connection of new hardware, and in some cases, to automatically install the required device driver(s). Specifically, Plug and Play, implemented by the PC's operating system, identifies new hardware devices connected to the PC and presents the user with installation options. Most new PCs are Plug and Play compatible.

In order for Plug and Play to function, the new hardware device must also be Plug and Play compatible. A major component of this compatibility is the Plug and Play device's identification (ID) stored in a memory location of the newly connected hardware device. This ID is provided to the PC during a Plug and Play handshake session. Plug and Play operates by requesting that a newly connected hardware device identify itself to the PC using the ID. In response, the device transmits its device ID to the PC which is then processed and identified by the PC's operating system. The signaling standard that governs the physical and electrical exchanges between the PC and the newly connected hardware device is the IEEE Std. 1284-1994 Standard Signaling Method for a Bi-directional Parallel Peripheral Interface for Personal Computers, published March 30, 1994.

The IEEE 1284 standard essentially upgrades the standard parallel port (SPP) interface that was designed for the IBM PC of the early 1980s. In particular, IEEE 1284 defines the physical and electrical interface between the PC and the peripheral device. The original SPP, referred to as the centronics style interface, provided an input/output (I/O) interface between a PC and a peripheral device at speeds on the order of 150 KBps. During this time frame, the fastest peripheral device was the dot matrix printer which could be easily accommodated by the SPP. The centronics style SPP, however, was unable to accommodate the much faster peripheral devices of modern personal computing, such as laser printers, which transfer color graphic files of sizes on the order of megabytes. An additional short-coming of the earlier parallel ports was the limitation of being able to transfer information in one direction only. Therefore, in the mid-1990s, the IEEE Standards Board approved the IEEE 1284 signaling method.

IEEE 1284 parallel ports run at 100 to 200 times faster than the original centronics-style parallel ports and not only facilitate faster communication, but also facilitate bi-directional communication, allowing the

printer to not only receive data, but also talk back to the PC. Today, most printer manufacturers design their newer generation printers to support the IEEE 1284 signaling standard. The device drivers that accompany these machines take full advantage of the high speed data rates and the bi-directional capability provided by IEEE 1284.

As shown in Fig. 1, IEEE 1284 defines five modes 15-19 of data transfer. The original centronics SPP included 17 signal lines and 8 ground lines. The signal lines included control lines for providing interface control, status lines for providing status indications for things such a paper empty signals, and data lines used to transfer data between the PC and the peripheral. IEEE 1284 uses these same data lines but redefines the electrical characteristics and signaling protocol required for communication. The first three data transfer modes of IEEE 1284, the compatibility mode 15, nibble mode 16, and byte mode 17, are provided mainly to support older uni-directional devices that continue to operate at the slower speeds. The Enhanced Parallel Port (EPP) mode 18 and Extended Capability Port (ECP) 19, facilitate operation of more modern, faster peripheral devices. EPP mode primarily supports CD ROMs, hard drives, and other non-printer peripherals and ECP mode is mainly directed towards printers and scanners. Microsoft® Windows® 95, for example, supports each of the five IEEE 1284 data transfer modes in its Plug and Play routine.

IEEE 1284 also defines the requirements of the device ID used, for example, in the Plug and Play implementation. As addressed above, the device ID is requested from the new hardware device by the PC in order for the PC to be able to properly identify and install the device drivers necessary for the new hardware to function correctly. In accordance with IEEE 1284, the device ID includes specific case sensitive key values. These values includes fields such as MANUFACTURER, and MODEL which are abbreviated by IEEE 1284 as MFG, and MDL respectively. The MFG identifies the device

manufacturer, and MDL provides a specific model number or nomenclature. Other Key fields exist, such as CLASS (CLS) and DESCRIPTION (DES), but are not as widely used. All ID key values must consist of American Standard Code for Information Interchange (ASCII) values of 32-127 (20h-7Fh) and are stored in the newly connected peripheral's configuration memory, such as ROM, or EEPROM, and stored in the form of an INF file. INF files typically contain all the data and information necessary to configure and operate the respective device.

With Plug and Play implemented, a user is then able to implement routines such as the Microsoft® Install New Hardware Wizard, and navigate through the procedures required for the user to complete installation of the device. The problem is, however, that Plug and Play is not adequate to handle installation of all devices in all scenarios. One of the scenarios where Plug and Play is least helpful is installing shared printers in a networked PC environment. Specifically, Plug and Play will not identify, for example, a new printer added to a print server operating in a Novell computer network environment.

Additionally, Plug and Play cannot presently apriorily determine whether a particular device installation will be problematic and thus cannot inform the user of any special measures necessary to resolve these problems in order to properly install the new hardware device. Therefore, a different technique is needed to resolve these types of problems and thus provide a user with the ability to successfully complete installation of a printer in a networked PC environment.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are incorporated in and constitute a part of the specification, illustrate embodiments of the invention and, together with the general description given above and the detailed description

of the embodiments given below, serve to explain the principles of the invention.

Fig. 1 is a chart illustrating each of the modes of IEEE Std. 1284.

Fig. 2 is a diagram of an exemplary PC with a scanner and printer attached.

Fig. 3 is a diagram of the PC of Fig. 2 networked with other PCs.

Fig. 4 is a diagram of the PC of Fig. 2 networked with other PCs and an additional printer.

Fig. 5 is a flow chart describing an exemplary method of implementing the present invention.

Fig. 6 is a flow chart of the ID read process of Fig. 5.

Fig. 7 is a flow chart of the translation/comparison process of Fig. 5.

Fig. 8 is an illustration of exemplary contents of a peripheral device's INF file, an example of a concatenated ID string, and an example of a host device's INF file contents.

Fig. 9 is a flow chart of the process of installing a previously installed printer at a different workstation.

DETAILED DESCRIPTION

An embodiment of the present application retrieves an IEEE 1284 device ID from any peripheral device that supports the IEEE 1284 signaling standard. It extracts manufacturer and model information from the device ID stored e.g., in a printer, and translates it into the manufacturer and model name used by the Microsoft® Windows® operating system, for example, in its printer selection process. The manufacturer and model names are then used in order to select the proper software driver required to complete the installation of the new hardware device.

In Fig. 2, a PC workstation 2 includes a stand-alone PC 5 with a peripheral device attached, such as a scanner 4. If a user desires to attach a

printer 6 to the PC 5, workstation 2B is created. In order to successfully connect the printer 6, an operating system used by the processor of the PC 5 may use a routine such as the Microsoft® Install New Hardware Wizard, previously described.

A user would initiate installation by first connecting the new hardware and then activating the Hardware Wizard routine. The routine would assist the user in identifying the newly connected printer 6, and provide the user with a list of potentially matching drivers from which to choose. A processor of the PC 5 would attempt to match the device ID of printer 6 to a list of device drivers stored in its own memory. If a match occurs, the device driver will be selected in order to facilitate installation. If no match is found, the user is then permitted to either provide a diskette containing the proper device driver, or select a driver from the list of drivers.

In Fig. 3, for example, PCs 5, 7, and 9 are connected to one another using a connection medium 8 to form a computer network 12. A printer server 10 is also connected to the network for the purpose of allocating the use of printer 6 between each of the PCs 5, 7, and 9. In this arrangement, the printer 6 is not directly attached to a PC but is attached to the print server 10 instead. In the configuration of Fig. 3, if the user desires to add an additional printer 11, as shown in Fig. 4, the Hardware Wizard cannot be used by the PC 5 user to install the additional printer 11. This routine cannot be used because, as stated earlier, printer 6 is not attached directly to a PC but is attached instead, to the print server 10. Similarly, new printer 11 is also be connected to the print server 10. Since the Hardware Wizard, Plug and Play, and other similar routines, are designed for stand alone PC operation, none of the PCs 5, 7, or 9, would be able to detect and select the proper driver for installation of newly added printer 11. In short, the print server 10 would not be able to automatically select a driver for the newly added printer 11 without the aid of some other device or method.

By providing, for example, the print server 10 and the PC 5, of Fig. 4, with an exemplary embodiment of the present invention, such as illustrated in selection/installation process 50 of Fig. 5, the user would be able to connect new hardware to the computer network 12 and be able to install the necessary drivers. The user is also informed, beforehand, whether a particular installation will present problems. Thus, by using the aforementioned exemplary embodiment of the present invention, the user can automatically select a device driver, such as a printer driver required for printer 11, and resolve these potential installation problems.

In Fig. 5, the operating system of PC 5 along with the present invention 50, would first recognize 100 that a new device has been installed on the output port of the print server 10. The user would then be prompted as to whether installation of the new device should be attempted 102. If installation is desired, the routine, partially stored in a memory of PC 5 and print server 10, would proceed to read 200 the IEEE 1284 device ID from the newly connected device, e.g., printer 11. If device installation is not desired, the process would tterminate 700.

In order for the print server 10 to read the IEEE 1284 device ID from a newly added peripheral, such as printer 11, the print server 10 must first transmit a read request 200A data signal, as shown in Fig. 6. Transmission of this signal will ultimately determine whether the printer 11 is IEEE 1284 compatible 200B. If compatibility is confirmed, the printer's device ID is retrieved 200C and stored in an appropriate memory location.

As addressed earlier, the device ID may be found in the memory of printer 11 as a string of ASCII characters between the values of 20h-7Fh, and may comprise key values such as MFG, MDL, CLS, and DES. The idea of the present invention is that the ID string, comprising at least the MFG and MDL key values, may be manipulated in a number of different ways to assist in identification of the drivers of a shared peripheral. However, in the

exemplary embodiment illustrated in Fig. 5, not all of the key values are used by selection/installation process 50.

When these fields are received and stored 200D, the routine 50 will then attempt to compare 300 the ID to a list of device drivers stored in an INF file of the PC 5. The invention may be implemented by manipulating any of the IEEE 1284 ID key fields and comparing these manipulated fields with stored device data. However, as shown in Fig. 7, the MFG and MDL key values are the primary key values used. First, upon retrieval, the MFG and MDL values are directly compared 300A with device names on a list of device drivers stored in PC 5. If a match 300B is found, the matching driver is selected 400. On the other hand if the comparison between the MFG and MDL key values with the listed names does not produce a match, the MFG and MDL key values are concatenated 300C, i.e., they are combined to form an ID string. The concatenated ID string is then compared 300D to the same list of drivers. Again, if a match 300E is found, the matching driver is selected 400. If a match 300E is not found, the MFG key value is then removed 300F, leaving the MDL as the remaining key value in the ID string. Finally, the MDL key value is compared with the list 300G in order to find a match 380. If a match 380 is found, the matching driver is selected. However, at this point, if a match is not found, the process will not be able to automatically select the appropriate driver. The translations that occur at 300C and 300F are necessary because the device ID codes, as stored in the printer 11, do not map directly into the device ID display strings associated with the device drivers. For example, Fig. 8 illustrates the manner in which typical MFG and MDL codes would appear in a file 30 stored in a memory of the printer 11. The MFG field may contain the value "printer-maker" minus the quotes to denote the manufacturer of the printer, as shown in Fig. 8. The MDL field may contain the value "PM Laserjet 3T" minus the quotes, to denote the specific model of the printer.

In the example of Fig. 8, after the MFG and MDL key fields have been retrieved and compared 300 with the list of drivers, the selection/installation process 50 will then concatenate the two key values 300B, producing "printer-makerPM_9876, 32." This value is then compared to names on the list of drivers stored in a file 34 of the user's PC. Although the MFG and MDL fields are utilized in the present invention, it is possible, as stated above, that other key field values may be used to create the device ID, such as a compatibility (CID) key field. The CID key field informs a Microsoft® Windows® based processor if the peripheral device will emulate other devices. When a match is finally achieved, a device driver is selected 400.

Again referring to Fig. 5, after selection of the device driver, the installation/selection routine 50 would determine if the particular peripheral device and the corresponding selected driver have historically created installation problems 500 for users. This feature is implemented by compiling known information into a data file, storing the information in database 501 in the print server 10, and making this information available to users. The selected driver is then compared with the database 501 to determine if the present installation is even possible, and/or determine if historical problems exist with respect to this particular installation. For example, some printers are considered to be host-based devices. That is, in order for these printers to initially be set-up for use, for example to have the printer heads properly configured and the printer's ink flow adjusted, they must be first attached to a local PC. A user would not be able connect such a host-based printer to a network without having first configured the printer by connecting it to a host PC. Additionally, some printers will simply not work unless they are connected directly to a host workstation. This type of information is particularly helpful to prevent the user from wasting time trying to accomplish device installations that will be problematic at best, if not impossible. This type of information is stored in database 501. Finally, the information is

presented to the user along with information required to resolve the installation problems 600, so that the device driver can be properly installed. The user is now able to install 650 the device driver and store information related to the particular installation in a memory of the print server 840. This process ends at 700 after the correct driver has been selected with the user able to install the selected driver in the PC 5.

In another embodiment of the present invention, subsequent users of workstations connected to network 12 may also desire to configure their PC 5 for installation of the newly connected printer 11. These users may benefit from previous installation information 840 of the routine 800, illustrated in Fig. 9. This information is stored after the first installation of a newly connected printer associated with print server 10 and network 12. That is, all the steps and information taken by the user that first installed printer 11, can be made available to all subsequent network users desiring to also install the proper device drivers in their PC in order to print to the same printer 11. Included, is information such as the specific driver(s) selected, the reference name of the printer, as well as other relevant information. Since this information is available to the user, the remainder of the installation process, for this particular user, can be simplified. Simplification is necessary because sometimes the 1284 codes are incomplete or inaccurate.

To implement the process, the routine 800, would first read the 1284 device IDs from the peripheral device, as executed in block 200 of Fig. 5. Next, it is determined 810, based upon the ID, whether a current desired installation is the first installation of the printer 11 involving the particular port of print server 10. If the current installation is the first time printer 11 has been installed at this particular port of the print server 10, i.e. the first installation, the process executes blocks 300, 380, 400, 500, and 600 from Fig. 5. This series of executed functions permit the user to install 820 printer 11 and store pertinent information, relating to this installation, in a memory of the

print server 840. On the other hand, if the current installation is not the first installation, i.e., this is not the first time printer 11 has been installed at the port of print server 10, the user will be presented with the information that was used to install the printer 11 during the earlier installation 830. This information is retrieved from a memory 840 of the print server 10.

In order to verify that printer 11 was indeed previously installed and not some other device, the IEEE 1284 ID of the previous installation, is retrieved from the stored information 840. If the IEEE 1284 ID of the current installation matches the ID of the earlier installation 850, confirmation is established that the current printer 11 was the device previously installed at the particular port of print server 10. In this case, the benefit of the previous user information, retrieved at 830, is used and installation of the printer 11 is completed by executing block 380 of process 50 and continuing. However, if the comparison of the IEEE 1284 IDs 850, did not produce a match, installation of the drivers would be completed based upon steps 400-600 of Fig. 5. If the IEEE 1284 IDs did not match, this would be an indication that the printer 11, had been changed since the previous installation of the device driver drivers. The user would then be permitted to select and install the required printer (driver) and then store relevant installation information in the database 840. This stored information is in-turn, made available to subsequent users.

The advantage of this process is that once one workstation has been configured to print to a particular printer, other users of workstations in the same office, connected to the network 12, will have simplified set-up procedures, based upon the information of the earlier installation.

From the invention thus described, it will be obvious that the invention may be varied in many ways. Such variations are not to be regarded as a departure from the spirit and scope of the invention, and all such modifications

as would be obvious to one skilled in the art are intended for inclusion within the scope of the following claims.